**Description**

Ractors can't access global variables, but some special global variables should be accessed. There are several types.

**Proposal**

(1) Read-only global variables

```
# process-local (readonly): other commandline parameters
'
- p' => $-p,
'-$l' => $-l,
'-$a' => $-a,

# process-local (readonly): getpid
'$$' => $$,
```

(2) scope local variables

```
# thread local: process result
'?' => $?,

# scope local: match
'-' => $-,inspect,
'&' => $&,
'-' => $-,
'\'' => $',
'+' => $+,           
'1' => 1,

# scope local: last line
'_' => $_,

# scope local: last backtrace
'@' => $@,
'!' => $!,
```

(3) Ractor local variables

```
# ractor-local (derived from created ractor): debug
'.DEBUG' => $DEBUG,
'-d' => $-d,

# ractor-local (derived from created ractor): verbose
'VERBOSE' => $VERBOSE,
'-w' => $-w,
'-W' => $-W,
'-v' => $-v,

# ractor local: stdin, out, err
'stdin' => stdin.inspect,
'stdout' => stdout.inspect,
'stderr' => stderr.inspect,
```
Implementation: https://github.com/ruby/ruby/pull/3670
I'll merge it soon.

Discussion

- only accessible from main ractor?
  - $0:
    - ARGV, ARGF, $.
- only accessible from main ractor because they will be obsolete
  - $, $/ $\$
- So difficult: $' / $LOADED_FEATURES and $: / $LOAD_PATH

Associated revisions

Revision 99310e3e - 10/20/2020 06:38 AM - ko1 (Koichi Sasada)
Some global variables can be accessed from ractors
Some global variables should be used from non-main Ractors.
[Bug #17268]

```ruby
# ractor-local (derived from created ractor): debug
'$$DEBUG' => $DEBUG,
'$$-d' => $$-d,

# ractor-local (derived from created ractor): verbose
'$$VERBOSE' => $$VERBOSE,
'$$-w' => $$-w,
'$$-W' => $$-W,
'$$-v' => $$-v,

# process-local (readonly): other commandline parameters
'$$-p' => $$-p,
'$$-l' => $$-l,
'$$-a' => $$-a,

# process-local (readonly): getpid
'$$' => $$,

# thread local: process result
'$$?' => $$?,

# scope local: match
'$$-' => $$-.inspect,
'$$a' => $$a,
'$$' => $$,
'$$\'' => $$',
'$$+' => $$+,
'$$1' => $$1,

# scope local: last line
'$$_' => $$_,

# scope local: last backtrace
'$$#' => $$#
'$$!' => $$!,

# ractor local: stdin, out, err
'$$stdin' => $$stdin.inspect,
'$$stdout' => $$stdout.inspect,
'$$stderr' => $$stderr.inspect,
```

History

#1 - 10/20/2020 01:49 AM - ko1 (Koichi Sasada)
- Description updated

#2 - 10/20/2020 06:39 AM - ko1 (Koichi Sasada)
- Status changed from Open to Closed

Applied in changeset git|99310e3eb56fbc85bb45119285812eb959449d0c.
Some global variables can be accessed from ractors

Some global variables should be used from non-main Ractors.

[Bug #17268]

```ruby
# ractor-local (derived from created ractor): debug
'$DEBUG' => $DEBUG,
'$-d' => $-d,

# ractor-local (derived from created ractor): verbose
'$VERBOSE' => $VERBOSE,
'$-w' => $-w,
'$-W' => $-W,
'$-v' => $-v,

# process-local (readonly): other commandline parameters
'$-p' => $-p,
'$-l' => $-l,
'$-a' => $-a,

# process-local (readonly): getpid
'$' => $,

# thread local: process result
'?' => $?,

# scope local: match
'-' => -.inspect,
'&' => &,
'\' => \/,
'\\' => \,
'+' => +,
'1' => 1,

# scope local: last line
'_' => _,

# scope local: last backtrace
'@' => @,
'!' => !,

# ractor local: stdin, out, err
'stdin' => stdin.inspect,
'stdout' => stdout.inspect,
'stderr' => stderr.inspect,
```

#3 - 10/22/2020 02:56 PM - Dan0042 (Daniel DeLorme)

Why can't ractors access global variables? Shouldn't it be like constants, where a frozen/shareable object can be accessed?

```ruby
$x = [1, 2, 3]
Ractor.new { $x }.take  # I understand that this can't work
$x = true
Ractor.new { $x }.take  # but why is this not ok?
```

#4 - 10/22/2020 07:31 PM - Eregon (Benoit Daloze)

I think this issue is mostly about "special" global variables, not so much about "regular" global variables.

I wonder if "regular" global variables should be Ractor-local, or behave like constants.

Since most of the above became Ractor-local (instead of per-Process before) or a narrower scope, I think Ractor-local would make most sense.

The only special variables that are still process-local are read-only, which means it's the same as if they were Ractor-local read-only.

#5 - 10/22/2020 09:18 PM - Dan0042 (Daniel DeLorme)

Eregon (Benoit Daloze) wrote in #note-4:

```
I wonder if "regular" global variables should be Ractor-local, or behave like constants.
```

I wouldn't mind either way. Although if Ractor-local I would hope they are derived from created ractor, like $DEBUG and $VERBOSE.

#6 - 10/26/2020 04:36 PM - ko1 (Koichi Sasada)
The proposal is accepted.

I wonder if "regular" global variables should be Ractor-local, or behave like constants.

This is rejected at first by Matz (my original proposal is making all $gvars are ractor-local), to avoid confusion.